

Non-Myopic Learning in Repeated Stochastic Games

Jacob W. Crandall

Department of Electrical Engineering and Computer Science,
Masdar Institute of Science and Technology
Abu Dhabi, UAE

Abstract

This paper addresses learning in repeated stochastic games (RSGs) played against unknown associates. Learning in RSGs is extremely challenging due to their inherently large strategy spaces. Furthermore, these games typically have multiple (often infinite) equilibria, making attempts to solve them via equilibrium analysis and rationality assumptions wholly insufficient. As such, previous learning algorithms for RSGs either learn very slowly or make extremely limiting assumptions about the game structure or associates' behaviors. In this paper, we propose and evaluate the notion of game abstraction by experts (Gabe) for two-player general-sum RSGs. Gabe reduces an RSG to a multi-armed bandit problem, which can then be solved using an expert algorithm. Gabe maintains many aspects of the original game, including security and Pareto optimal Nash equilibria. We demonstrate that Gabe substantially outperforms existing algorithms in many scenarios.

1 Introduction

Many scenarios require computers, robots, and other intelligent devices to repeatedly interact with people and other machines. In these interactions, associates' behaviors are often not known *a priori*. For example, in the context of Ad Hoc team coordination (Stone et al. 2010), a machine must be able to effectively interact with arbitrary teammates without prior coordination. These teammates are created by other stakeholders, and may not share all the same objectives or run the same algorithms. Such interactions occur in new-age power systems, electronic commerce, and other domains.

Many repeated interactions can be formulated as repeated stochastic games (RSGs). Because such interactions are finite, a successful player must learn effective strategies within short time scales. Unfortunately, this is extremely challenging. Existing learning algorithms for this domain either require thousands of interactions to learn (Claus and Boutilier 1998; Crandall and Goodrich 2011) or make narrow assumptions about associates' behaviors (and, thus, learn ineffective strategies when these assumptions fail).

These deficiencies are due to a number of characteristics of the domain. RSGs are punctuated by large strategy spaces, even for games with relatively few states. Furthermore, the strategies of associates are unknown *a priori*, and can change over time (when they also learn). Since

these games typically have multiple (often infinite) equilibria, associates' future behaviors can be very difficult to infer or learn – reliance on so-called rationality and equilibrium computation is insufficient.

Game abstraction has emerged in recent years as an effective methodology for dealing with games having large state and action spaces (Gilpin and Sandholm 2006; Schnitzlein, Bowling, and Szafron 2009; Ganzfried, Sandholm, and Waugh 2012; Sandholm and Singh 2012). In most approaches, the game is abstracted to a smaller game. An equilibrium strategy is then computed for this smaller game, but is then executed in the original game. While this abstraction methodology is effective in large zero-sum, extensive-form games, it does little to help an algorithm deal with multiple equilibria and adaptive associates.

We introduce a new game-abstraction method called *game abstraction by experts* (Gabe). Gabe reduces an RSG to a multi-armed bandit problem, which can then be solved using an expert algorithm such as Exp3 (Auer et al. 1995), UCB (Auer, Cesa-Bianchi, and Fischer 2002), or EEE (de Farias and Megiddo 2004). It does this by computing a finite set of expert strategies or learning rules, which are followed based on the selections made by the expert algorithm. The effectiveness of the resulting algorithm depends on both the set of expert strategies and the ability of the expert algorithm to select the best experts for the given scenario.

In this paper, we describe a general set of experts for two-player general-sum RSGs. While this set of experts effectively reduces the set of strategies that a player can play, it maintains important characteristics of the original game, such as its security level and various Pareto optimal Nash equilibria (NEs). We demonstrate that, when combined with an effective expert algorithm, Gabe outperforms many other algorithms in three two-player RSGs played against a variety of both non-learning and learning associates. In short, Gabe provides a mechanism for agents to quickly learn profitable non-myopic strategies that are otherwise difficult to learn.

2 Notation

We consider two-player RSGs played by players i and $-i$. An RSG consists of a set of *stage games* (or states) S . In each state $s \in S$, both players choose an action from a finite set. Let $A(s) = A_i(s) \times A_{-i}(s)$ be the set of joint actions available in s , where $A_i(s)$ and $A_{-i}(s)$ are the action sets of players i

and $-i$, respectively. Each *episode* (or *round*) of an RSG begins in the start state $\hat{s} \in S$ and terminates when some goal state $s_g \in G \subseteq S$ is reached. Once a goal state is encountered, a new episode begins in state \hat{s} .

When joint action $\mathbf{a} = (a_i, a_{-i})$ is played in state s , each player receives the finite rewards $r_i(s, \mathbf{a})$ and $r_{-i}(s, \mathbf{a})$, respectively. The world also transitions to some new state s' with probability defined by $P_M(s, \mathbf{a}, s')$. We assume that the transition model P_M and the reward functions $r_i(s, \mathbf{a})$ and $r_{-i}(s, \mathbf{a})$ are known to both players before the game begins, and that the players can observe each other's actions.

3 Example RSG

For illustrative purposes, we consider a Microgrid Scenario (Figure 1) involving two players. These players interact for an unspecified number of days, each seeking to maximize their own utility over this time. To gain utility, a player executes its tasks within the specified time windows (Figure 1a-b). These tasks require the specified electricity loads. A task can be executed no more than once a day, and must be executed within a single time period.

The players share a limited electricity supply, which has the per-hour generation characteristics shown in Figure 1(c). We assume that generation occurs at the beginning of each hour. Additionally, the system automatically stores up to five units of unused electricity for later use. For simplicity, we assume that all storage is lost at the end of each day.

If the players attempt to consume more electricity in an hour than is available, a blackout occurs. In a blackout, electricity storage empties, and the tasks that the players attempted to execute are left unfinished (as if they had not been initiated). A cost of two utility units is incurred to bring the microgrid back online, which is evenly distributed among the players that attempted to execute tasks in that hour.

In this RSG, the state of the world is defined by (1) the current time (hour), (2) the amount of stored electricity, and (3) the set of currently active tasks (i.e., unexecuted tasks whose time windows correspond to the current hour). In total, this game has 2,033 unique states. The beginning state is the state of the system in hour $t = 0$. The goal states consist of all states with time $t = 24$. Each player's action set for each state is the set of all subsets of its active tasks.

There is not sufficient electricity generated in a day for all tasks to be executed. Furthermore, some high-value tasks can only be executed when the other player complies. For example, there is only sufficient electricity for Player 2 to execute Task 12 if Player 1 refrains from executing Task 1 prior to time $t = 6$. Similarly, Player 1 can only successfully execute Task 10 if Player 2 refrains from Task 20. The players must coordinate and compromise to be successful.

Our observations are that existing learning algorithms are unable to learn effective strategies in this RSG, especially when they are paired with each other. For example, model-free reinforcement learning (RL; e.g., Q-learning (Watkins and Dayan 1992)) is unable to learn effective strategies within a reasonable number of days due to the game's large state-joint action space. Existing algorithms that learn at faster time scales still fail to learn collaborative

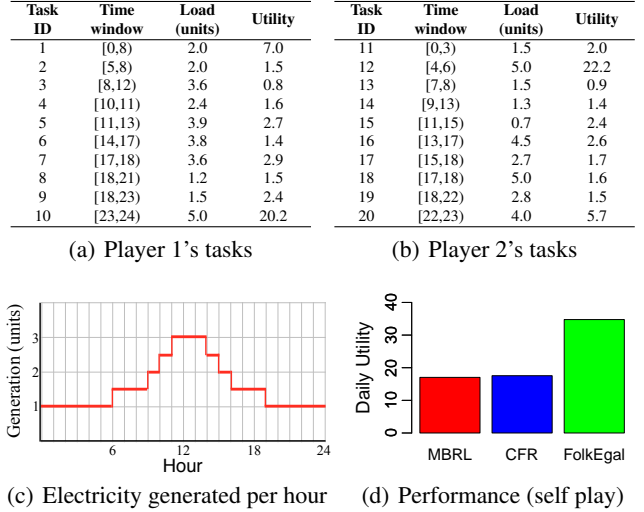


Figure 1: The Microgrid Scenario.

solutions. For example, when both players utilize model-based RL (MBRL) or counterfactual regret (CFR, an effective algorithm for the poker domain (Zinkevich et al. 2007; Johanson et al. 2012)), they fail to learn a collaborative solution in self play. As a result, as shown in Figure 1(d), their average daily utilities are far below those obtained by FolkEgal (de Cote and Littman 2008). On the other hand, FolkEgal does not perform effectively when associating with algorithms that do not meet its assumptions (see Table 1(a)).

Because RSGs are punctuated with large state-action spaces, multiple (often infinite) equilibria, and associates with unknown (potentially changing) strategies, it is infeasible to learn non-myopic solutions using algorithms that use locally optimal computations and assumptions. While such methods might be sufficient if all consequences of the repeated interactions could be fully represented, the state-action spaces of such representations are prohibitively large. Novel game-abstraction methods are needed. In this paper, we study game abstraction by experts (Gabe).

4 Game Abstraction By Experts (Gabe)

Gabe (summarized in Algorithm 1) reduces the strategy space of an RSG to a finite set Φ of strategies or algorithms. Each $\phi \in \Phi$ defines a policy for all states $s \in S$. In this way, Gabe converts an RSG into a multi-armed bandit problem, where each *arm* is an expert $\phi \in \Phi$. Thus, rather than learning a separate policy for each state $s \in S$, the agent must learn which high-level strategies or algorithms $\phi \in \Phi$ are the most profitable, and then follow these strategies.

Gabe has similarities to the work of Elidrisi et al. (2014), in which the RSG is reduced to a normal-form game by identifying common sequences of actions (or paths) for each player. These paths are then used as the actions of a normal-form game. Identifying these actions relies on effective exploration strategies and clustering and thresholding algorithms. Gabe instead defines Φ to reduce the RSG to a multi-armed bandit problem. In so doing, it maintains important

Algorithm 1 Game abstraction by experts (Gabe).

Input: An expert algorithm \mathcal{A}
Initialize: Compute a set Φ of experts
Run: \mathcal{A} on the set Φ
 In each round t
 - Select an expert $\phi_t \in \Phi$ using \mathcal{A}
 - Follow the strategy prescribed by ϕ_t throughout round t
 - Update each $\phi \in \Phi$ and \mathcal{A} as specified

attributes of the original RSG.

Algorithm 1 requires that we solve two technical problems. First, we must identify an expert algorithm \mathcal{A} that learns effectively in repeated interactions with unknown associates. A common goal for such algorithms is to learn to play nearly as well as the best expert would have performed had it always been followed. This problem has been well-studied in the literature (Auer et al. 1995; Bowling 2004; Arora, Dekel, and Tewari 2012; Crandall 2014; Cesa-Bianchi, Dekel, and Shamir 2013).

Second, we must define the set of experts Φ . Ideally, in any scenario (i.e., RSG and associate) that a player is likely to encounter, at least one $\phi \in \Phi$ should play near-optimally. However, no single expert need be effective in all situations.

Given that expert algorithms have been well-studied, we focus in the next section on identifying and computing a general set of experts Φ for two-player general-sum RSGs.

5 Experts

Our set Φ for two-player RSGs consists of three genres of algorithms. The first two genres, *leader* and *follower strategies*, were defined by Littman and Stone (2001). We introduce the third genre, call *preventative strategies*, in this paper. These three genres define appropriate responses to many algorithms that associates are likely to use.

Leader Strategies

A leader strategy tries to encourage its associate to follow a *target solution* (Littman and Stone 2005). The leader plays its own portion of the target solution as long as its associate plays its part. However, when the associate deviates from this solution, the leader retaliates in subsequent moves to ensure that the associate does not profit from the deviation.

To derive an effective set of leader strategies, we must do three things. First, we must efficiently compute desirable target solutions. Second, we must select a set of target solutions, one solution corresponding to each leader strategy we place in Φ . Third, we must determine how to punish deviations from the target solution.

Computing Target Solutions Possible target solutions of an RSG can be computed by solving a Markov decision process (MDP). This MDP is defined by P_M , the RSG’s joint-action space, and a payoff function that is a convex combination of the players’ rewards (de Cote and Littman 2008). That is, for $\omega \in [0, 1]$, the payoff function is given by

$$\sigma_\omega(s, \mathbf{a}) = \omega r_i(s, \mathbf{a}) + (1 - \omega) r_{-i}(s, \mathbf{a}). \quad (1)$$

ω	$V_i^\omega(s), V_{-i}^\omega(s)$	Description of possible target solution
0.1, 0.2	11.3, 40.0	(5,12), (7,1), (10,4), (11,14), (12,5), (13,16), (14,15), (15,17), (17,18), (21,19), (22,20)
0.3	36.8, 32.7	(5,12), (7,1), (7,13), (10,4), (11,14), (12,5), (13,16), (14,15), (15,17), (17,7), (19,19), (22,9), (23,10)
0.4	38.3, 31.8	(5,12), (6,1), (7,2), (10,4), (11,14), (12,5), (13,16), (14,15), (15,17), (17,7), (19,9), (21,19), (23,10)
0.5	39.8, 30.3	(5,12), (6,1), (7,2), (10,4), (12,14), (12,5), (13,15), (14,16), (16,17), (17,7), (20,8), (22,9), (23,10)
0.6, 0.7	41.2, 28.6	(5,12), (6,1), (7,2), (10,4), (11,14), (12,5), (13,16), (14,15), (15,6), (17,7), (20,8), (22,9), (23,10)
0.8, 0.9	42.0, 26.3	(5,12), (6,2), (7,1), (10,4), (11,3), (12,5), (14,6), (14,15), (16,17), (17,7), (20,8), (22,9), (23,10)

Table 1: Possible target solutions computed in the Microgrid Scenario. Descriptions specify sequences of 2-tuples of the form $(time, taskID)$ indicating when tasks are executed.

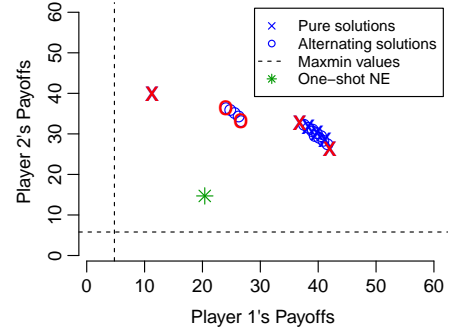


Figure 2: Payoff vectors of possible target solutions in the Microgrid Scenario. Red indicates selection by our method.

Then, the value for taking joint-action \mathbf{a} in state s is

$$Q^\omega(s, \mathbf{a}) = \sigma_\omega(s, \mathbf{a}) + \sum_{s' \in S} P_M(s, \mathbf{a}, s') V^\omega(s'), \quad (2)$$

where $V^\omega(s') = \max_{\mathbf{a} \in A(s')} Q^\omega(s', \mathbf{a})$, which can be solved in polynomial time using linear programming (Papadimitriou and Tsitsiklis 1987; Littman, Dean, and Kaelbling 1995).

Let $MDP(\omega)$ denote the joint strategy produced by solving the MDP for a particular ω . Also, let $V_i^\omega(s)$ be player i ’s expected future payoff from state s when $MDP(\omega)$ is followed. Then, the ordered pair $(V_i^\omega(s), V_{-i}^\omega(s))$ is the joint payoff vector for the target solution defined by $MDP(\omega)$. This payoff vector is Pareto optimal.

By varying ω , we can compute a variety of possible target solutions. For example, Table 1 gives the solutions computed for the Microgrid Scenario using this method. We call these solutions *pure solutions*. Additional possible target solutions can be obtained by alternating between solutions produced using different ω . We call these solutions *alternating solutions*. For example, the players could play $MDP(0.1)$ in odd-numbered rounds, and $MDP(0.3)$ in even-numbered rounds. This would produce the average joint payoff vector of $(24.05, 36.35)$ in the Microgrid Scenario. In this paper, we only include alternating cycles of length 2, since longer cycles are likely to be difficult to reinforce using a leader strategy, and hence will likely be difficult to agree upon.

Which Target Solutions? Figure 2 shows the joint payoff vectors of possible pure and alternating target solutions in the Microgrid Scenario. As with many other RSGs, the one-shot NEs (one shown) are Pareto dominated by several possible target solutions. Furthermore, the possible target solutions in which each player’s payoff exceeds its maximin strategy can be sustained as NEs of the repeated game (Gintis 2000). As such, these possible target solutions offer a variety of potentially desirable equilibrium solutions, each differing in the value provided to each player.

In practice, more experts makes the task of finding the best expert more difficult for the expert algorithm. Thus, we form leader experts for only up to five of the possible target solutions we have identified. First, we select the solution that maximizes the minimum payoff between the players (i.e., the egalitarian solution). Next, we select the two points that give each player the highest payoff while providing the other player with at least its security level. The last two points are chosen to maximize the Euclidean distance from the other selected payoff vectors. For the Microgrid Scenario, our algorithm selects the target solutions shown in red in Figure 2.

Adding punishment When the associate deviates from the target solution, the leader plays an attack strategy (usually its minimax strategy (Littman and Stone 2005)) in subsequent moves of the game. In practice, we deviate slightly from this protocol: player i only punishes deviations by player $-i$ that substantially lower i ’s payoffs. Formally, let s_τ denote the state of the τ th move of the round. Then, player i begins punishing $-i$ when its action deviates from the target solution and the following two conditions hold:

$$(1) V_i^\omega(s_\tau) + r_i^t(\tau - 1) < V_i^\omega(s_{\tau-1}), \quad (3)$$

$$(2) \sum_{j=1}^{\tau} r_i^t(j) + V_i^\omega(s_\tau) < \alpha_i^t, \quad (4)$$

where $r_i^t(j)$ is i ’s payoff after the j th move of round t . Also, $\alpha_i^t = \lambda \alpha_i^{t-1} + (1 - \lambda) R_i^t$, $\lambda \in (0, 1)$, and R_i^t is i ’s total payoff in round t . α_i^0 is set to i ’s payoff in the egalitarian solution.

When player $-i$ deviates and conditions (1-2) hold, i punishes $-i$ for the remainder of round t , and continues to do so in subsequent rounds until $-i$ ’s payoffs are at least δ less than they would have been had $-i$ not deviated. We chose this punishment mechanism because RSGs are inherently noisy in practice. The associate’s payoffs are often uncertain, and the reward and transition functions of RSGs can be non-deterministic. In such situations, being more lenient helps avoid cycles of unnecessary punishment.

Since the attack strategy can also be computed in polynomial time, a leader strategy can be built in polynomial time.

Follower Strategies

We also include follower strategies in Φ . Rather than attempt to drive play toward a particular solution, followers seek to maximize their payoffs against the strategy they attribute to their associate. We use followers that estimate their associate’s strategy in three different ways. The first set of followers assume their associate is playing a leader strategy. Against such associates, a player maximizes its payoffs by

playing its part of the corresponding target solution. Thus, we form a separate follower strategy for each of the selected target solutions mentioned earlier, each of which plays its part of its target solution unconditionally.

The set Φ also includes two other follower strategies: its maximin strategy, which is a best response to an associate seeking to minimize its payoffs, and MBRL, which plays a best response to stationary associates it can model. MBRL estimates its associate’s strategy using the Fictitious-play assessment (Fudenberg and Levine 1998). That is, in state s , i assumes that $-i$ plays action a_{-i} proportionally to the frequency it has played it in past visits to s . We denote this probability as $\gamma_{-i}(s, a_{-i})$.

Preventative Strategies

Due to their representations of the world, some learning associates have difficulty perceiving the punishment signals communicated by a leader strategy. As such, they do not learn to play the target solution intended by the leader. In such cases, *preventative strategies* can sometimes be more effective. Rather than punishing past deviations, preventative strategies seek to make deviations unprofitable in first place. They do this by anticipating profitable deviations the associate might make, and then acting in advance so that such deviations are unprofitable to the associate.

We include one preventative algorithm in Φ , which we call Bouncer. Bouncer seeks to minimize the difference between the players’ payoffs, without further regard for its own payoffs. Formally, Bouncer computes both $Q_i(s, \mathbf{a})$ and $Q_{-i}(s, \mathbf{a})$ using SARSA (Rummery and Niranjan 1994), where the estimated strategies for the players are given by the Fictitious-play assessments $\gamma_i(s, a_i)$ and $\gamma_{-i}(s, a_{-i})$, respectively. It then selects actions as follows:

$$a_i^*(s) = \min_{a_i \in A_i(s)} \sum_{a_{-i} \in A_{-i}(s)} \gamma_{-i}(s, a_{-i}) U(s, (a_i, a_{-i})), \quad (5)$$

where $U(s, \mathbf{a}) = |Q_i(s, \mathbf{a}) - Q_{-i}(s, \mathbf{a})|$.

6 Properties of the Abstraction

The set Φ as defined in Section 5 is an abstraction of the strategy space of the original RSG. This abstraction limits the set of strategies available to a player. However, it preserves the follow three attributes of the original RSG.

Property 1 (ϵ -Pareto optimal NEs) Φ contains leader strategies that seek to play ϵ -Pareto optimal NEs of the repeated game. These same NEs exist in the original RSG.

Property 2 (Security) Because Φ contains the maximin strategy, it maintains the original RSG’s security level.

Property 3 (Best response) Φ contains MBRL, which learns a best response (in the original game) to the play of stationary associates having the same state representation.

Thus, Φ preserves important attribute of the original RSG, attributes that correspond to previously stated goals for learning in games (Powers and Shoham 2005). In the next section, we show that this simplified strategy space allows an expert algorithm to learn non-myopic strategies in many scenarios, which helps it to outperform existing algorithms.

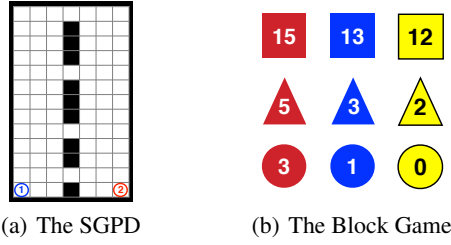


Figure 3: Two RSGs.

7 Empirical Performance

We combined the expert algorithms Exp3 (Auer et al. 1995) and S++ (Crandall 2014) with the set Φ to form two new algorithms: Gabe-Exp3 and Gabe-S++. Exp3 is a commonly used expert algorithm, while S++ is a recently developed algorithm that has demonstrated high empirical performance in repeated normal-form games. We evaluated these algorithms against ten associates in three separate games.

Associates We evaluated Gabe-Exp3 and Gabe-S++ when paired with both non-learning and learning algorithms. The non-learning algorithms were Coop, Bully, FolkEgal (de Cote and Littman 2008), Maximin, Bouncer, and CFR-NE. Coop plays the strategy that maximizes its associate’s payoffs. Bully is identical to FolkEgal except that it seeks to enforce the target solution that maximizes its own payoffs subject to the other player receiving at least its security level. CFR-NE computes an ϵ -NE prior to the game using counterfactual regret in simulated self play (Zinkevich et al. 2007; Johanson et al. 2012). The learning algorithms were MBRL, CFR, Gabe-Exp3, and Gabe-S++. CFR is identical to CFR-NE except that it also updates its strategy after each round based on its associate’s observed actions.

Games We evaluated the algorithms in three RSGs: the Microgrid Scenario, the SGPD (Figure 3(a)), and a block game. These games require the players to make and accept different kinds of compromises to be successful. The SGPD (fully explained by Goodrich, Crandall, and Stimpson, 2003) is a prisoners’ dilemma in maze form with 26,896 states. A player receives 30 points when it moves to the other player’s start position, and loses one point every time it attempts a move. A successful player in this RSG must effectively balance cooperation and defection.

The Block Game is a turn-taking game in which the two players share a set of blocks (Figure 3(b)). In each round, the players take turns selecting blocks (the oldest player always goes first). A round ends once both players have three blocks. If a player’s blocks form a valid set (i.e., all blocks of the same color, all blocks of the same shape, or none of the blocks have the same color or shape), her payoff is the sum of the numbers on her blocks. Otherwise, her payoff is the sum of the numbers divided by 4. The sub-game perfect one-shot NEs of this game give each player 18 points. However, these solutions are Pareto dominated by the solution in which the players alternate between taking all of the squares and all of the triangles (which gives each player an average

payoff of 25). Even better, a player could potentially bully the other by always insisting on taking all of the squares.

Results Figure 4 provides a summary of the average empirical performance of the four learning algorithms in each game over the first 500 rounds. Table 2 shows the results of each pairing in each game in 365-round games. Averaged over all pairings, Gabe-S++ was the highest performer in each game. The order of the other players varied from game to game. The consistently high performance of Gabe-S++ can be traced to the two components of Gabe: the strategy abstraction Φ and the quality of the expert algorithm \mathcal{A} .

First, the strategy abstraction Φ allows Gabe-S++ to (1) make profitable compromises with associates that are apt to compromise, (2) exploit naive associates, and (3) avoid being exploited by antagonistic associates. For example, FolkEgal, Gabe-S++, and Bouncer (in the SGPD only) are apt to play the egalitarian solution in each game. Gabe-S++ is able to learn to play these solutions against these associates (and, hence, performs well) because at least one of the leader and follower strategies in Φ play these compromises. Φ also contains Bouncer, which allows it to learn mutual cooperation against MBRL in the SGPD in some trials. Meanwhile, the follower strategies in Φ permit Gabe-S++ to exploit Coop and to avoid being perpetually exploited by Bully, CFR, and CFR-NE in each game. Thus, Φ contains a set of strategies that allow it to perform effectively against each of these ten associates in each game.

MBRL and CFR, on the other hand, cannot effectively represent the egalitarian solution, or other effective non-myopic compromises. Thus, while they can exploit naive algorithms (such as Coop) and also avoid being exploited, they do not typically perform well against associates that are apt to cooperate in these games. This is because they cannot represent non-myopic strategies in general-sum RSGs.

Second, Gabe-S++ performs well because of its expert algorithm. While Gabe-Exp3 used the same set of expert strategies Φ as Gabe-S++, it did not perform nearly as well in any of these three games. Despite sometimes sharing theoretical properties (such as no-regret), both the short- and long-term empirical performance of expert algorithms can vary substantially in repeated interactions (Crandall 2014). Thus, both the selection of the expert algorithm and the strategy set Φ are important components of Gabe.

8 Conclusion and Discussion

In this paper, we have proposed game abstractions by experts (Gabe) for two player, repeated stochastic games (RSGs). Gabe reduces an RSG to a multi-armed bandit problem, which can then be solved using an expert algorithm. Gabe maintains many important characteristics of the original game, including security, best response, and Pareto optimality. Furthermore, we demonstrated empirically that, given an effective expert algorithm, Gabe outperforms existing learning algorithms in a number of general-sum RSGs.

Gabe differs from previous game-abstraction methods (Gilpin and Sandholm 2006; Schnizlein, Bowling, and Szafron 2009; Ganzfried, Sandholm, and Waugh 2012; Sandholm and Singh 2012). Whereas previous methods seek

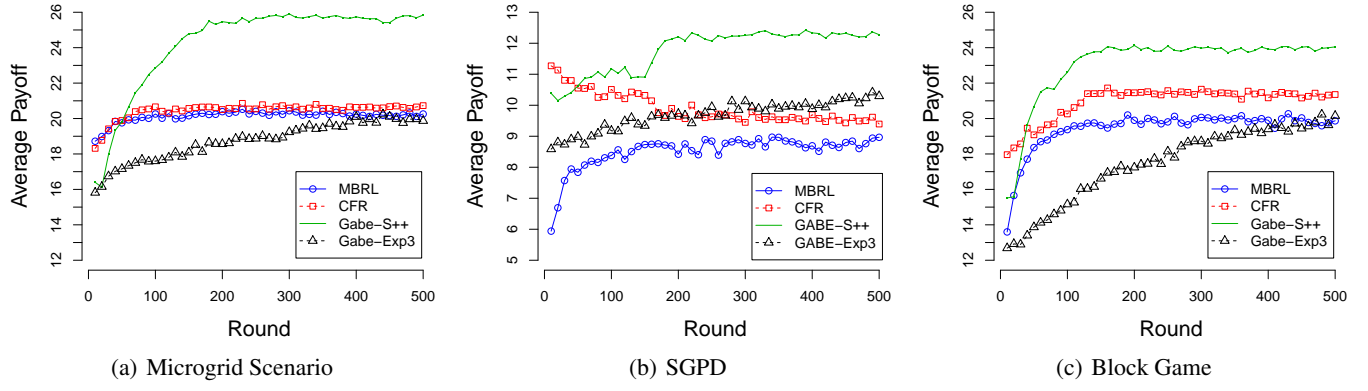


Figure 4: Average payoffs over time against all associates in each game.

(a) Microgrid Scenario

Algorithm	Associate										Ave.
	Gabe-S++	Gabe-Exp3	MBRL	CFR-NE	CFR	FolkEgal	Bully	Bouncer	Maximin	Coop	
Gabe-S++	33.01	24.79	19.57	14.00	21.86	34.15	16.36	13.24	22.03	39.10	23.81
CFR	22.36	25.43	17.67	17.55	17.55	19.11	11.98	15.07	19.24	37.89	20.38
MBRL	21.75	25.27	16.95	16.33	16.58	21.87	13.29	14.80	18.29	35.94	20.11
Coop	23.48	25.50	21.17	10.03	20.79	23.95	18.80	12.00	11.06	33.60	20.04
Gabe-Exp3	21.17	23.75	13.78	11.67	13.54	25.75	11.72	10.87	15.26	35.57	18.31
CFR-NE	19.15	24.59	17.40	17.55	17.55	18.14	10.72	8.92	16.64	30.74	18.14
FolkEgal	33.83	22.28	13.59	6.36	17.83	34.75	8.81	7.97	9.89	24.74	18.00
Maximin	15.05	20.10	16.23	16.31	15.95	23.93	11.01	16.03	16.76	26.20	17.76
Bully	31.14	16.18	14.54	1.74	9.71	5.14	2.75	6.02	10.62	41.00	13.88
Bouncer	11.47	10.37	12.60	6.71	12.02	7.08	5.41	1.20	14.16	11.76	9.28

(b) SGPD

Algorithm	Associate										Ave.
	Gabe-S++	Gabe-Exp3	MBRL	CFR-NE	CFR	FolkEgal	Bully	Bouncer	Maximin	Coop	
Gabe-S++	15.69	12.88	10.02	5.86	5.86	15.94	5.93	15.92	5.86	22.00	11.60
Bouncer	15.92	13.56	13.87	7.00	7.00	16.00	7.01	7.00	7.00	16.00	11.04
FolkEgal	15.86	12.90	5.74	4.66	4.66	16.00	4.72	16.00	4.66	16.00	10.12
CFR-NE	9.34	12.05	9.94	7.00	7.00	12.01	7.06	7.00	7.00	22.00	10.04
CFR	9.34	12.05	9.94	7.00	7.00	12.01	7.06	7.00	7.00	22.00	10.04
Maximin	9.34	12.05	9.94	7.00	7.00	12.01	7.06	7.00	7.00	22.00	10.04
Bully	9.29	12.15	9.63	6.97	6.97	12.01	7.02	7.01	6.97	19.00	9.70
Gabe-Exp3	12.34	11.43	6.78	4.61	4.61	13.93	4.61	13.56	4.61	18.59	9.51
MBRL	8.53	9.26	6.11	3.49	3.49	11.18	3.98	13.79	3.49	21.00	8.43
Coop	0.00	9.09	0.16	0.00	0.00	16.00	8.00	16.00	0.00	16.00	6.53

(c) Block Game

Algorithm	Associate										Ave.
	Gabe-S++	Gabe-Exp3	MBRL	CFR-NE	CFR	FolkEgal	Bully	Bouncer	Maximin	Coop	
Gabe-S++	23.10	23.25	18.51	15.61	18.47	24.61	7.84	16.32	40.00	40.00	22.77
CFR	17.86	25.06	18.02	18.00	18.00	15.48	6.50	17.70	31.51	39.95	20.81
MBRL	14.73	21.05	15.72	15.13	15.41	16.54	7.26	15.35	35.60	36.24	19.27
CFR-NE	17.23	24.48	17.94	18.00	18.00	9.52	-4.97	15.23	34.46	36.23	18.61
Bully	29.99	24.05	21.00	1.83	30.46	9.83	-5.38	-5.70	26.83	40.00	17.29
Gabe-Exp3	13.06	18.88	13.54	9.38	12.80	18.26	4.48	11.62	29.90	34.71	16.66
FolkEgal	24.57	20.46	16.90	4.91	12.77	24.96	-0.22	0.03	16.61	32.50	15.35
Coop	10.00	13.45	10.22	10.92	10.02	17.50	10.00	18.00	12.64	25.00	13.77
Bouncer	16.43	11.81	15.94	15.25	17.70	0.93	-4.61	17.71	12.17	18.00	12.13
Maximin	2.80	4.76	3.20	4.73	4.29	5.12	3.59	12.76	11.95	6.48	5.97

Table 2: Average per-round payoffs (over 25) trials for all pairings in (a) the the Microgrid Scenario, (b) the SGPD, and (c) the Block Game. Each trial consisted of 365 rounds. Each cell shows the row algorithm’s payoffs when paired with the column algorithm.

to reduce the number of states and actions in the game (to make equilibrium computation feasible), Gabe abstracts the high-level strategies of the game. This allows it to represent non-myopic strategies. Since these abstraction methods have different purposes, Gabe can be used simultaneously with previous game-abstraction methods where necessary.

References

- [Arora, Dekel, and Tewari 2012] Arora, R.; Dekel, O.; and Tewari, A. 2012. Online bandit learning against an adaptive adversary: from regret to policy regret. In *Proceedings of the 29th International Conference on Machine Learning*, 1503–1510.
- [Auer et al. 1995] Auer, P.; Cesa-Bianchi, N.; Freund, Y.; and Schapire, R. E. 1995. Gambling in a rigged casino: the adversarial multi-armed bandit problem. In *Proceedings of the 36th Symposium on the Foundations of Computer Science*, 322–331.
- [Auer, Cesa-Bianchi, and Fischer 2002] Auer, P.; Cesa-Bianchi, N.; and Fischer, P. 2002. Finite-time analysis of the multi-armed bandit problem. *Machine Learning* 47:235–256.
- [Bowling 2004] Bowling, M. 2004. Convergence and no-regret in multiagent learning. In *Advances in Neural Information Processing Systems 17*, 209–216.
- [Cesa-Bianchi, Dekel, and Shamir 2013] Cesa-Bianchi, N.; Dekel, O.; and Shamir, O. 2013. Online learning with switching costs and other adaptive adversaries. In *Advances in Neural Information Processing Systems 26*, 1160–1168.
- [Claus and Boutilier 1998] Claus, C., and Boutilier, C. 1998. The dynamics of reinforcement learning in cooperative multiagent systems. In *Proceedings of the 15th National Conference on Artificial Intelligence*, 746–752.
- [Crandall and Goodrich 2011] Crandall, J. W., and Goodrich, M. A. 2011. Learning to compete, coordinate, and cooperate in repeated games using reinforcement learning. *Machine Learning* 82(3):281–314.
- [Crandall 2014] Crandall, J. W. 2014. Towards minimizing disappointment in repeated games. *Journal of Artificial Intelligence Research* 49:111–142.
- [de Cote and Littman 2008] de Cote, E. M., and Littman, M. L. 2008. A polynomial-time Nash equilibrium algorithm for repeated stochastic games. In *Proceedings of the 24th Conference on Uncertainty in Artificial Intelligence*, 419–426.
- [de Farias and Megiddo 2004] de Farias, D., and Megiddo, N. 2004. Exploration–exploitation tradeoffs for expert algorithms in reactive environments. In *Advances in Neural Information Processing Systems 17*, 409–416.
- [Elidrisi et al. 2014] Elidrisi, M.; Johnson, N.; Gini, and Crandall, J. W. 2014. Fast adaptive learning in repeated stochastic games by game abstraction. In *Proceedings of the 13th International Conference on Autonomous Agents and Multi-agent Systems*.
- [Fudenberg and Levine 1998] Fudenberg, D., and Levine, D. K. 1998. *The Theory of Learning in Games*. The MIT Press.
- [Ganzfried, Sandholm, and Waugh 2012] Ganzfried, S.; Sandholm, T.; and Waugh, K. 2012. Strategy purification and thresholding: Effective non-equilibrium approaches for playing large games. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems*.
- [Gilpin and Sandholm 2006] Gilpin, A., and Sandholm, T. 2006. A competitive Texas Hold’em poker player via automated abstraction and real-time equilibrium computation. In *Proceedings of the National Conference on Artificial Intelligence*.
- [Gintis 2000] Gintis, H. 2000. *Game Theory Evolving: A Problem-Centered Introduction to Modeling Strategic Behavior*. Princeton University Press.
- [Goodrich, Crandall, and Stimpson 2003] Goodrich, M. A.; Crandall, J. W.; and Stimpson, J. R. 2003. Neglect tolerant teaming: Issues and dilemmas. In *AAAI Spring Symposium on Human Interaction with Autonomous Systems in Complex Environments*.
- [Johanson et al. 2012] Johanson, M.; Bard, N.; Lanctot, M.; Gibson, R.; and Bowling, M. 2012. Efficient Nash equilibrium approximation through Monte Carlo counterfactual regret minimization. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems*, 837–846.
- [Littman and Stone 2001] Littman, M. L., and Stone, P. 2001. Leading best-response strategies in repeated games. In *IJCAI workshop on Economic Agents, Models, and Mechanisms*.
- [Littman and Stone 2005] Littman, M. L., and Stone, P. 2005. A polynomial-time Nash equilibrium algorithm for repeated games. *Decision Support Systems* 39:55–66.
- [Littman, Dean, and Kaelbling 1995] Littman, M. L.; Dean, T. L.; and Kaelbling, L. P. 1995. On the complexity of solving markov decision problems. In *Proceedings of the 11th International Conference on Uncertainty in Artificial Intelligence*.
- [Papadimitriou and Tsitsiklis 1987] Papadimitriou, C. H., and Tsitsiklis, J. N. 1987. The complexity of Markov chain decision processes. *Mathematics of Operations Research* 12(2):441–450.
- [Powers and Shoham 2005] Powers, R., and Shoham, Y. 2005. Learning against opponents with bounded memory. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence*, 817–822.
- [Rummery and Niranjan 1994] Rummery, G. A., and Niranjan, M. 1994. On-line Q-learning using connectionist systems. Technical Report CUED/F-INFENG-TR 166, Cambridge University, UK.
- [Sandholm and Singh 2012] Sandholm, T., and Singh, S. 2012. Lossy stochastic game abstraction with bounds. In *Proceedings of the 13th ACM Conference on Electronic Commerce*, 880–897.

- [Schnizlein, Bowling, and Szafron 2009] Schnizlein, D.; Bowling, M.; and Szafron, D. 2009. Strategy purification and thresholding: Effective non-equilibrium approaches for playing large games. In *Proceedings of the International Joint Conference on Artificial Intelligence*.
- [Stone et al. 2010] Stone, P.; Kaminka, G. A.; Kraus, S.; and Rosenschein, J. S. 2010. Ad hoc autonomous agent teams: Collaboration without pre-coordination. In *Proceedings of the Twenty-Fourth Conference on Artificial Intelligence*, 1504–1509.
- [Watkins and Dayan 1992] Watkins, C. J., and Dayan, P. 1992. Q-learning. *Machine Learning* 8:279–292.
- [Zinkevich et al. 2007] Zinkevich, M.; Bowling, M.; Johanson, M.; and Piccione, C. 2007. Regret minimization in games with incomplete information. In *Neural Information Processing Systems*.